Name: Mohd Isa Jaffar *Student id:* 20011527 *Section:* 1 *Serial#:* 15

QUESTION #	1	2	3	4	TOTAL
MAX POINTS	20	18	16	28	
POINTS EARNED	17	16.5	06	27	66.5

UNIVERSITY OF BAHRAIN — COLLEGE OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SCIENCE — TIME: 90 MINUTES
ITCS241: ASSEMBLY LANGUAGE PROGRAMMING — SECOND TEST — DATE: DEC 27, 2010

QUESTION ONE: {20 pts}

1) Given an array: `FF SWORD 750 dup(?)`; Write instructions to store in EAX and EBX the sum of negative and positive values in FF correspondingly. Keep array FF UNCHANGED.

```
MOV ECX, lengthof FF
XOR EAX,EAX
XOR EBX,EBX
MOV edi,0
L1: CMP FF[edi],0
JNL NEG
MOVSX EBX, FF[edi]
ADD EBX,EDX
jmp next
NEG: MOVSX EDX,FF[edi]
ADD EAX,EDX
next: ADD edi,2
LOOP L1
```

2) Assume R1 and R2 are predefined unsigned words, If R1 < R2 then calculate EDI= 16*flags and swap the last\2 double words pushed onto the stack and else calculate R2=R2/32 and clear the odd-numbered bits in R2. ***MUL,IMUL,DIV,IdIV not allowed.***

```
MOV AX, R1          other : SHR AX, 5
CMP R1,AX           MOV R2,AX
JAE other           AND R2, 5555h
PushF               Done :
Pop DI
MOVSX EDI,DI
SHL EDI,4
Pop EDX
Pop ECX
Push EDX
Push ECX
jmp DONE
```

Name: *Student id:* *Section:* *Serial=:*

QUESTION TWO: Write a sequence of instructions to perform each of the following 5 tasks:

1) Give ONE instruction to clear the left half in a double word pointed by `esi` register. **{2 pts}**

```
AND [ESI], 0000FFFFh
```

2) Give ONE instruction to set the sign bit in BX. Keep other bits unchanged. **{2 pts}**

```
OR BX, 8000h
```

3) Give no More than 4 instructions to shift the entire value in **EBX:EAX** 12 bits to the left **{4 pts}**

```
MOV ECX, 12
L1: SHL EAX, 1
    RCL EBX, 1
    Loop L1
```

4) Give no more than 6 instructions to multiply the last 2 words pushed onto the stack and replace them by the resulting product. **{5 pts}**

```
Pop AX
Pop BX
MUL BX
Push DX
Push AX
```

5) Give no more than 5 instructions to store in h the sum of the 2 halves of ebx register. **{5 pts}**

```
XOR EAX, EAX
SHLD EAX, EBX, 16
MOVSX EBX, BX
ADD EAX, EBX
MOV h, EAX
```

Name: *Student id:* *Section:* *Serial#:*

QUESTION THREE: What will be in the specified registers after executing the following code? **{ 20 pts}**

a)
```
MOV   AX, 5A50H
MOV   BX, 7C4FH
MUL   AH
```
AX = 1944 H

b)
```
MOV   AX, 303AH
MOV   CX, 20H
IDIV  CL
```
AX = 4CD H

c)
```
MOV   AX, 768CH
TEST  AX, 8C30H
XOR   AX, 0FF0H
```
AX = 797C H

d)
```
MOV   AX, 2F08H
MOV   BX, 794CH
SHLD  AX, BX, 4
```
AX = 7907 H

e)
```
MOV   AX, 6750H
MOV   BX, 3EFFH
OR    AX, BX
ROL   AX, 4
RCR   BX, 1
```
AX = FFF7 H

BX = 9F7F H

f)
```
MOV   AX, 6750H
MOV   BX, 3FFFH
MOV   CL, 4
SAR   AX, CL
NEG   BX
```
AX = 0675 H

BX = C001 H

g)
```
MOV   AX,3FFFH
MOV   BX,6750H
CMP   AL, AH
JG    L1
INC   AL
JMP   L2
L1:INC  AH
L2:
```
AX = 40FF H

h)
```
MOV   AX,3FFFH
MOV   BX,6750H
CMP   AL, AH
JA    L3
INC   BL
JMP   L4
L3:INC  BH
L4:
```
AX = 6761 H

Name: *Student id:* *Section:* *Serial#:*

QUESTION FOUR: Implement the following C++ program in Assembly language **{28 pts}**

```
#include <iostream>
using namespace std;

void func (int [],int,int&,int&);// define proc FUN4 so that it can be invoked!

void main()
{     int t[10]={ 9,2,3,4,5,6,7,13,11,9};  // define int type using sword!!
      int fard, zaj ;
      func (t,10, &fard, &zaj);
      cout << fard << '\t' << zaj << endl ;  //9 is ASCII code for tab
}
void func (int D[], int m, int *f, int *z)
{
      *f = 0; *z = 0;
      for (int k = 0; k < m ; k++)
         if (D[k] % 2 != 0 )
             (*f)++ ;
         else
             (*z)++ ;
}
```

```
INCLUDE Irvine32.inc
.data
Sword t 9,2,3,4,5,6,7,13,11,9
Sword Fard 0
Sword Zaj 0
.CODE
Func PROC USES ESI ECX EBX EDI EAX, Arr:PTR sword, Size:dword, Fa:PTR sword, Za:PTR sword
   MOV ESI , Arr          ; Pointer to array
   MOV ECX, Size          ; Pointer to size
   MOV [ecx]              ; loop count
   MOV ebx, Fa
   MOV edx, Za
   MOV edi, 0
   L1: popAd
      MOV AX, sword PTR [esi + edi]
      MOV CL,2
      IDIV CL
      CMP AH, 0
      je IncreZa
      INC sword PTR [ebx]
      jmp DONE
      IncreZa : INC sword PTR [edx]
      DONE : popAd
```

Name: Student id: Section: Serial#:

```
ADD EDI, 2
Loop L1
ret
func ENDP
Main PROC

Invoke func ADDR t, ADDR C, ADDR fard, ADDR zaj

MOVSX EAX, fard
CALL WriteInt
MOV AL, 9
CALL WriteChar

MOVSX EAX, zaj
CALL WriteInt
MOV AL, 9
CALL WriteChar

CALL CRLF
exit
Main ENDP
END Main
```